Kernel RX-Algorithm: A Nonlinear Anomaly Detector for Hyperspectral Imagery

Heesung Kwon, *Member, IEEE,* and Nasser M. Nasrabadi, *Fellow, IEEE*

***Abstract*—In this paper, we present a nonlinear version of the well-known anomaly detection method referred to as the RX-algorithm. Extending this algorithm to a feature space associated with the original input space via a certain nonlinear mapping function can provide a nonlinear version of the RX-algorithm. This nonlinear RX-algorithm, referred to as the kernel RX-algorithm, is basically intractable mainly due to the high dimensionality of the feature space produced by the nonlinear mapping function. However, in this paper it is shown that the kernel RX-algorithm can easily be implemented by *kernelizing* the RX-algorithm in the feature space in terms of *kernels* that implicitly compute dot products in the feature space. Improved performance of the kernel RX-algorithm over the conventional RX-algorithm is shown by testing several hyperspectral imagery for military target and mine detection.**

***Index Terms*—Anomaly detection, hyperspectral images, kernel-based learning, kernels, target detection.**

I. Introduction

RECENTLY, there has been a major interest in using hyperspectral imagery (HSI) for anomaly and target detection [1]–[9]. Hyperspectral imagery provides significant information about the spectral characteristics of the materials in the scene. Typically, a hyperspectral spectrometer provides hundreds of narrow contiguous bands which can be exploited to detect and identify certain types of materials in the image. Hyperspectral sensors that exploit the reflective (or emissive) properties of objects can collect data in the visible and short-wave infrared (IR) regions (or the mid-wave and long-wave IR regions) of the spectrum. Collection of this data allows the algorithm to detect and identify targets of interest in a hyperspectral scene by exploiting the spectral signature of the materials.

The process of detecting and identifying a target in hyperspectral imagery can be considered as consisting of two-stages. The first stage is an anomaly detector which identifies spectral anomalies or a localized spectral difference. The second stage identifies whether or not the anomaly is a target or natural clutter. This stage can be achieved if the spectral signature of the target is known which can be obtained from a spectral library or a spectral matched filter designed from a set of training data [1], [5].

Anomaly detectors are pattern recognition schemes that are used to detect objects that might be of military interest. Almost all anomaly detectors attempt to locate anything that looks different spatially or spectrally from its surroundings. In spectral anomaly detection algorithms, pixels (materials) that have a significantly different spectral signature from their neighboring background clutter pixels are identified as spectral anomalies. Spectral anomaly detection algorithms [8], [10]–[13] could also use spectral signatures to detect anomalies embedded within background clutter with a very low signal-to-noise ratio (SNR). In spectral anomaly detectors, no prior knowledge of the target spectral signature is utilized or assumed.

Most of the detection algorithms in the literature [1], [3], [8], [13] assume that the HSI data can be represented by the multivariate normal (Gaussian) distribution and under the Gaussianity assumption, the generalized-likelihood ratio test (GLRT) is used to test the hypotheses to find the existence of a target in the image. The Gaussianity assumption has been used mainly because of mathematical tractability that allows the formation of widely used detection models, such as GLRT. However, in reality the HSI data might not closely follow the Gaussian distribution. Nevertheless, in various fields of signal processing, GLRT is used to detect signals (targets) of interest in noisy environments.

In [3], a spectral anomaly detection algorithm was developed for detecting targets of unknown spectral distribution against a background with unknown spectral covariance. This algorithm is now commonly referred to as the RX anomaly detector which has been successfully applied to many hyperspectral target detection applications [8], [12], [14]. It is now considered as the benchmark anomaly detection algorithm for multispectral/hyperspectral data. The RX-algorithm is a constant false-alarm rate (CFAR) adaptive anomaly detector which is derived from the generalized-likelihood ratio test. In the RX-algorithm, anomaly detection is formulated as two hypotheses, $\mathbf{H}_0$ and $\mathbf{H}_1$. The first hypothesis $\mathbf{H}_0$ models the background as a Gaussian distribution with zero mean and an unknown background covariance matrix which is estimated locally or globally from the data. The second hypothesis $\mathbf{H}_1$ models the target as a linear combination of a target signature and background noise. So, under $\mathbf{H}_1$ a spectral vector is represented by a Gaussian distribution with a mean equal to the signature of the target and an additive noise equal to the background covariance matrix in hypothesis $\mathbf{H}_0$.

The RX-algorithm is based on exploiting the difference between the spectral signatures of an input pixel and its surrounding neighbors. This distance comparison is very similar to the Mahalanobis distance measure which is done by comparing the corresponding wavelengths (spectral bands) of two measurements. The conventional RX distance measure does not take into account the higher order relationships between the spectral bands at different wavelengths. The nonlinear relationships between different spectral bands within the target or clutter spectral signature need to be exploited in order to

Manuscript received February 27, 2004; revised October 10, 2004.

The authors are with the U.S. Army Research Laboratory, Adelphi, MD 20783-1197 USA (e-mail: hkwon@arl.army.mil; nnasraba@arl.army.mil).

Digital Object Identifier 10.1109/TGRS.2004.841487

better distinguish between the two hypotheses. Furthermore, the Gaussian assumption in the RX-algorithm for the distributions of the two hypotheses $\mathbf{H}_0$ and $\mathbf{H}_1$ in general is not valid. Therefore, in this paper we formulate a nonlinear version of the RX-algorithm by assuming a Gaussian distribution for the two hypotheses in a high-dimensional feature space obtained by a nonlinear mapping. Modeling the input data in the feature space by a Gaussian distribution is equivalent to representing the distribution of the input data with a much more complex model when defined in the original input space.

Transforming each spectral pixel into a high-dimensional feature space (which could be potentially of infinite dimension) by a nonlinear mapping will result in a spectral pixel in a feature space consisting of possibly the original spectral bands and many nonlinear combinations of the spectral bands of the original spectral signature. This way the higher order correlations between the spectral bands are exploited by implementing the RX-algorithm in the feature space. However, the nonlinear RX-algorithm cannot be implemented directly due to the high dimensionality of the feature space. It is shown in Section IV that because the RX-algorithm consists of inner products of spectral vectors, it is possible to implement a kernel-based nonlinear version of the RX-algorithm by using kernel functions, and their properties [15]. In fact, by using kernel functions, no explicit knowledge of the actual nonlinear mapping is necessary which means the RX-algorithm is not computed explicitly in the feature space. This property is the major advantage of the kernel-based methods that reduce a nonlinear algorithm to a linear one in some high-dimensional feature space.

Kernel-based versions of a number of feature extraction or pattern recognition algorithms have recently been proposed [16]–[22]. In [19], a kernel version of principal component analysis (PCA) was proposed for nonlinear feature extraction and in [20] a nonlinear kernel version of the Fisher discriminant analysis was implemented for pattern classification. In [21], a kernel-based clustering algorithm was proposed and in [17] kernels were used as generalized dissimilarity measures for classification. Kernel methods have also been applied to face recognition in [16]. In [22], a kernel version of a matched subspace detector is proposed for target detection in hyperspectral imagery.

This paper is organized as follows. Section II provides an introduction to the RX-algorithm. Section III describes kernel functions and their relationship with the dot product of input vectors in the feature space. In Section IV, we show the derivation of the kernel version of the RX-algorithm. In Section V, we describe how the kernel and conventional RX-algorithms are implemented for hyperspectral target detection applications. Experimental results comparing the RX-algorithm and kernel RX-algorithm are given in Section VI. Finally, in Section VII conclusion and discussion are provided.

II. Introduction to the RX-Algorithm

Reed and Yu [3] developed a GLRT, so-called RX anomaly detection, for HSI data assuming that the spectrum of the received signal (spectral pixel) and the covariance of the background clutter are unknown. In the conventional RX-algorithm, a nonstationary local mean is subtracted from each spectral pixel. The local mean μ is obtained by sliding a double concentric window (a small inner window centered within a larger outer window) over every spectral pixel in the image and calculate the mean of the spectral pixels falling within the outer window. The size of the inner window is assumed to be the size of the typical target of interest in the image. The residual signal after mean subtraction is assumed to approximate a zero-mean pixel-to-pixel independent Gaussian random process.

Let each input spectral signal consisting of J spectral bands be denoted by $\mathbf{x}(n) = (x_1(n), x_2(n), \ldots, x_J(n))^T$. Define $\mathbf{X}_b$ to be a $J \times M$ matrix of the M reference background clutter pixels (or pixels in the outer window). Each observation spectral pixel is represented as a column in the sample matrix $\mathbf{X}_b$

$$\mathbf{X}_b = [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(M)]. \tag{1}$$

The two competing hypotheses that the RX-algorithm must distinguish are given by

$$\begin{aligned} \mathbf{H}_0 &: \mathbf{x} = \mathbf{n} \quad \text{(Target absent)} \\ \mathbf{H}_1 &: \mathbf{x} = a\mathbf{s} + \mathbf{n} \quad \text{(Target present)} \end{aligned} \tag{2}$$

where $a = 0$ under $\mathbf{H}_0$ and $a > 0$ under $\mathbf{H}_1$, respectively. $\mathbf{n}$ is a vector that represents the background clutter noise process, and $\mathbf{s}$ is the spectral signature of the signal (target) given by $\mathbf{s} = (s_1, s_2, \ldots, s_J)^T$. The target signature $\mathbf{s}$ and background covariance $\mathbf{C}_b$ are assumed to be unknown. The model assumes that the data arise from two normal probability density functions with the same covariance matrix but different means. Under $\mathbf{H}_0$, the data (background clutter) are modeled as $\mathcal{N}(0, \mathbf{C}_b)$, and under $\mathbf{H}_1$ the data are modeled as $\mathcal{N}(\mathbf{s}, \mathbf{C}_b)$. It should be pointed out that a key assumption in the RX-algorithm is that the background and target have the same covariance matrix in the model as shown in (2). In general, this is not a valid model if a particular target structure is to be detected. A more appropriate model would have two different covariance structures—one for anomaly (which could be target or background clutter) and one for background. However, the covariance structure for the anomaly cannot be estimated, since the statistical structure of the anomaly cannot be defined. Therefore, the same covariance structure for anomaly and background is adopted.

Assuming a single pixel target $\mathbf{r}$ as the observation test vector, the results of the RX-algorithm is given by

$$\begin{aligned} \mathrm{RX}(\mathbf{r}) = (\mathbf{r} - \hat{\boldsymbol{\mu}}_b)^T \Bigg(\frac{M}{M+1} \hat{\mathbf{C}}_b \\ + \frac{1}{M+1} (\mathbf{r} - \hat{\boldsymbol{\mu}}_b)(\mathbf{r} - \hat{\boldsymbol{\mu}}_b)^T \Bigg)^{-1} (\mathbf{r} - \hat{\boldsymbol{\mu}}_b) \underset{H_0}{\overset{H_1}{\gtrless}} \eta \end{aligned} \tag{3}$$

where η is a threshold of the test, $\hat{\mathbf{C}}_b$ is the background covariance matrix estimated from the reference background clutter data, and $\hat{\boldsymbol{\mu}}_b$ is the estimated background clutter sample mean. As $M \to \infty$, the RX-algorithm converges to

$$\mathrm{RX}(\mathbf{r}) = (\mathbf{r} - \hat{\boldsymbol{\mu}}_b)^T \hat{\mathbf{C}}_b^{-1} (\mathbf{r} - \hat{\boldsymbol{\mu}}_b). \tag{4}$$

Equation (4) is the RX expression that is implemented and is referred to as the conventional RX-algorithm in this paper.

III. Feature Space and Kernel Methods

Suppose the input hyperspectral data is represented by the data space ($\mathcal{X} \subseteq \mathcal{R}^J$) and $\mathcal{F}$ be a feature space associated with $\mathcal{X}$ by a nonlinear mapping function Φ

$$\begin{aligned} \Phi : \mathcal{X} &\rightarrow \mathcal{F} \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}) \end{aligned} \tag{5}$$

where $\mathbf{x}$ is an input vector in $\mathcal{X}$ that is mapped into a potentially much higher (could be infinite) dimensional feature space. Now, any linear algorithm can be remodeled into this high-dimensional feature space by replacing the original input data $\mathbf{X}_b := [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(M)]$ with the mapped data $\Phi(\mathbf{X}_b) = [\Phi(\mathbf{x}(1)), \Phi(\mathbf{x}(2)), \ldots, \Phi(\mathbf{x}(M))]$.

Due to the high dimensionality of the feature space $\mathcal{F}$, it is computationally not feasible to directly implement any algorithm in the feature space. However, kernel-based learning algorithms use an effective kernel trick to implement dot products in feature space by employing kernel functions [15]. Using the kernel trick (6), it allows us to implicitly compute the dot products in $\mathcal{F}$ without mapping the input vectors into $\mathcal{F}$; therefore, in the kernel methods, the mapping Φ does not need to be identified. The kernel representation for the dot products in $\mathcal{F}$ is expressed as

$$\begin{aligned} k(\mathbf{x}_i, \mathbf{x}_j) &= \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \\ &= \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j). \end{aligned} \tag{6}$$

Equation (6) shows that the dot products in $\mathcal{F}$ can be avoided and replaced by a kernel function k, a nonlinear function that can be easily calculated without identifying the nonlinear map Φ. A commonly used kernel is the Gaussian radial basis function (RBF) kernel: $k(\mathbf{x}, \mathbf{y}) = \exp((-\|\mathbf{x}-\mathbf{y}\|^2)/(c))$ where $c > 0$ is a constant. See [15] for detailed information about the properties of different kernels and kernel-based learning.

IV. Kernel RX-Algorithm

In this section, we formulate an *RX-like* anomaly detector in the feature space. It uses the same assumptions used in the RX-algorithm, i.e., the mapped input data in the feature space now consists of two Gaussian distributions, thus modeling the two hypotheses as

$$\begin{aligned} &\mathbf{H}_{0_\Phi} : \Phi(\mathbf{x}) = \mathbf{n}_\Phi \quad \text{(Target absent)} \\ &\mathbf{H}_{1_\Phi} : \Phi(\mathbf{x}) = b\Phi(\mathbf{s}) + \mathbf{n}_\Phi \quad \text{(Target present)} \end{aligned} \tag{7}$$

where $b = 0$ under $\mathbf{H}_{0_\Phi}$, and $b > 0$ under $\mathbf{H}_{1_\Phi}$, respectively. $\Phi(\mathbf{s})$ represents target spectral signature and $\mathbf{n}_\Phi$ represents a noise process in the feature space. The above linear model in the feature space has an equivalent nonlinear model in the input space, which is not exactly the same as the nonlinearly mapped version of the additive model given in (2).

The corresponding RX-algorithm in the feature space is now represented as

$$\text{RX}(\Phi(\mathbf{r})) = \left(\Phi(\mathbf{r}) - \hat{\boldsymbol{\mu}}_{b_\Phi}\right)^T \hat{\mathbf{C}}_{b_\Phi}^{-1} \left(\Phi(\mathbf{r}) - \hat{\boldsymbol{\mu}}_{b_\Phi}\right) \tag{8}$$

where $\hat{\mathbf{C}}_{b_\Phi}$ and $\hat{\boldsymbol{\mu}}_{b_\Phi}$ are the estimated covariance and mean of the background clutter samples in the feature space, respectively, given by

$$\hat{\mathbf{C}}_{b_\Phi} = \frac{1}{M} \sum_{i=1}^{M} \left(\Phi(\mathbf{x}(i)) - \hat{\boldsymbol{\mu}}_{b_\Phi}\right) \left(\Phi(\mathbf{x}(i)) - \hat{\boldsymbol{\mu}}_{b_\Phi}\right)^T \tag{9}$$

$$\hat{\boldsymbol{\mu}}_{b_\Phi} = \frac{1}{M} \sum_{i=1}^{M} \Phi(\mathbf{x}(i)). \tag{10}$$

The expression for the RX-algorithm in the feature space (8) was obtained by using the GLRT and assuming that both the background and target have a Gaussian distribution in the feature space with the same covariance matrix but with a different mean. The Gaussian assumption in the feature space allows the use of GLRT to obtain (8), which is equivalent a nonlinear RX-algorithm or a nonlinear GLRT procedure in the original input space.

A. Kernelization of the RX-Algorithm in the Feature Space

The RX-algorithm in the feature space given by (8) cannot be implemented explicitly due to the nonlinear mapping Φ, which produces a data space of high dimensionality. In order to avoid implementing (8) directly, we need to kernelize it by using the kernel trick introduced in Section III.

The estimated background covariance matrix can be represented by its eigenvector decomposition or so-called spectral decomposition [23] as given by

$$\hat{\mathbf{C}}_{b_\Phi} = \mathbf{V}_\Phi \boldsymbol{\Lambda}_b \mathbf{V}_\Phi^T \tag{11}$$

where $\boldsymbol{\Lambda}_b$ is a diagonal matrix consisting of the eigenvalues and $\mathbf{V}_\Phi$ is a matrix whose columns are the eigenvectors of $\hat{\mathbf{C}}_{b_\Phi}$ in the feature space

$$\mathbf{V}_\Phi = \left[\mathbf{v}_\Phi^1, \mathbf{v}_\Phi^2, \ldots, \mathbf{v}_\Phi^M\right] \tag{12}$$

where M is the maximum number of the eigenvectors with nonzero eigenvalues.

The pseudoinverse of the estimated background covariance matrix can also be written in terms of its eigenvectors and nonzero eigenvalues as

$$\hat{\mathbf{C}}_{b_\Phi}^{\#} = \mathbf{V}_\Phi \boldsymbol{\Lambda}_b^{-1} \mathbf{V}_\Phi^T. \tag{13}$$

Each eigenvector $\mathbf{v}_\Phi^j$ in the feature space, as shown in Appendix I, can be expressed as a linear combination of the centered input vectors $\Phi_c(\mathbf{x}(i)) = \Phi(\mathbf{x}(i)) - \hat{\boldsymbol{\mu}}_{b_\Phi}$ in the feature space

$$\mathbf{v}_\Phi^j = \sum_{i=1}^{M} \beta_i^j \Phi_c(\mathbf{x}(i)) = \mathbf{X}_{b_\Phi} \boldsymbol{\beta}^j \tag{14}$$

where $\mathbf{X}_{b_\Phi} = [\Phi_c(\mathbf{x}(1)), \Phi_c(\mathbf{x}(2)), \ldots, \Phi_c(\mathbf{x}(M))]$ and $\boldsymbol{\beta}^j = (\beta_1^j, \beta_2^j, \ldots, \beta_M^j)^T$, $j = 1, \ldots, M$, are the eigenvectors of the centered kernel matrix (Gram matrix) $\hat{\mathbf{K}}(\mathbf{X}_b, \mathbf{X}_b)$ normalized by the square root of their corresponding eigenvalues, as shown in Appendix I. For all the eigenvectors with nonzero eigenvalues

$$\mathbf{V}_\Phi = \mathbf{X}_{b_\Phi}\mathcal{B} \tag{15}$$

where $\mathcal{B} = (\boldsymbol{\beta}^1, \boldsymbol{\beta}^2, \ldots, \boldsymbol{\beta}^M)^T$.

Substituting (15) into (13) yields

$$\hat{\mathbf{C}}^{\#}_{b_\Phi} = \mathbf{X}_{b_\Phi}\mathcal{B}\boldsymbol{\Lambda}_b^{-1}\mathcal{B}^T\mathbf{X}_{b_\Phi}^T. \tag{16}$$

Inserting (16) into (8) it can be rewritten as

$$\mathrm{RX}(\Phi(\mathbf{r})) = \left(\Phi(\mathbf{r}) - \hat{\boldsymbol{\mu}}_{b_\Phi}\right)^T \mathbf{X}_{b_\Phi}\mathcal{B}\boldsymbol{\Lambda}_b^{-1}\mathcal{B}^T\mathbf{X}_{b_\Phi}^T \left(\Phi(\mathbf{r}) - \hat{\boldsymbol{\mu}}_{b_\Phi}\right). \tag{17}$$

The dot product term $\Phi(\mathbf{r})^T\mathbf{X}_{b_\Phi}$ in the feature space can be represented in terms of the kernel function

$$\begin{aligned}\Phi(\mathbf{r})^T\mathbf{X}_{b_\Phi} &= \Phi(\mathbf{r})^T([\Phi(\mathbf{x}(1)), \Phi(\mathbf{x}(2)), \ldots, \Phi(\mathbf{x}(M))] \\ &\quad - \frac{1}{M}\sum_{i=1}^{M}\Phi(\mathbf{x}(i))) \\ &= (k(\mathbf{x}(1), \mathbf{r}), k(\mathbf{x}(2), \mathbf{r}), \ldots, k(\mathbf{x}(M), \mathbf{r})) \\ &\quad - \frac{1}{M}\sum_{i=1}^{M} k(\mathbf{x}(i), \mathbf{r}) \\ &= \mathbf{k}(\mathbf{X}_b, \mathbf{r})^T - \frac{1}{M}\sum_{i=1}^{M} k(\mathbf{x}(i), \mathbf{r}) \equiv \mathbf{K}_\mathbf{r}^T \end{aligned}\tag{18}$$

where $\mathbf{k}(\mathbf{X}_b, \mathbf{r})^T$ represents a vector whose entries are the kernels $k(\mathbf{x}(i), \mathbf{r}), i = 1, \ldots, M$, and $(1/M)\sum_{i=1}^{M} k(\mathbf{x}(i), \mathbf{r})$ represents the scalar mean of $\mathbf{k}(\mathbf{X}_b, \mathbf{r})^T$. Similarly,

$$\begin{aligned}\hat{\boldsymbol{\mu}}_{b_\Phi}^T\mathbf{X}_{b_\Phi} &= \frac{1}{M}\sum_{i=1}^{M}\Phi(\mathbf{x}(i))^T \\ &\quad \times ([\Phi(\mathbf{x}(1)), \Phi(\mathbf{x}(2)), \ldots, \Phi(\mathbf{x}(M))] \\ &\quad - \frac{1}{M}\sum_{i=1}^{M}\Phi(\mathbf{x}(i))) \\ &= \frac{1}{M}\sum_{i=1}^{M}(k(\mathbf{x}(i), \mathbf{x}(1)), \\ &\quad \times k(\mathbf{x}(i), x(2)), \ldots, k(\mathbf{x}(i), \mathbf{x}(M))) \\ &\quad - \frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M} k(\mathbf{x}(i), \mathbf{x}(j)) \\ &= \frac{1}{M}\sum_{i=1}^{M}\mathbf{K}(\mathbf{x}(i), \mathbf{X}_b) - \frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M} k(\mathbf{x}(i), \mathbf{x}(j)) \\ &\equiv \mathbf{K}_{\hat{\boldsymbol{\mu}}_b}^T \end{aligned}\tag{19}$$

where $(1/M^2)\sum_{i=1}^{M}\sum_{j=1}^{M} k(\mathbf{x}(i), \mathbf{x}(j))$ represents the mean of all the entries of the Gram matrix $\mathbf{K}(\mathbf{X}_b, \mathbf{X}_b)$. The above transformations (18) and (19) are usually referred to as the empirical kernel map in the machine learning literature [15].

Also using the properties of the Kernel PCA, as shown in Appendix I, we have the relationship

$$\hat{\mathbf{K}}_b^{-1} = \mathcal{B}\boldsymbol{\Lambda}_b^{-1}\mathcal{B}^T \tag{20}$$

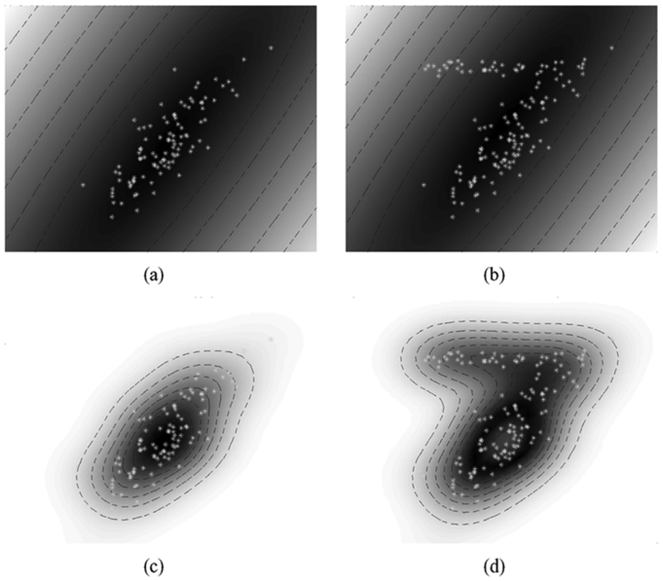

Fig. 1. Contour and surface plots of the RX-algorithm and Kernel RX-algorithm for two illustrative toy distributions: the conventional RX-algorithm for (a) a single Gaussian and (b) a mixture of Gaussians and the Kernel RX-algorithm for (c) a single Gaussian and (d) a mixture of Gaussians.

where $\hat{\mathbf{K}}_b$ denotes the centered Gram matrix. Substituting (18)–(20) into (17), the kernelized version of the RX-algorithm is given by

$$\mathrm{RX}_\mathbf{K}(\mathbf{r}) = \left(\mathbf{K}_\mathbf{r}^T - \mathbf{K}_{\hat{\boldsymbol{\mu}}_b}^T\right)^T \hat{\mathbf{K}}_b^{-1}\left(\mathbf{K}_\mathbf{r}^T - \mathbf{K}_{\hat{\boldsymbol{\mu}}_b}^T\right) \tag{21}$$

which can now be implemented with no knowledge of the mapping function Φ. The only requirement is a good choice for the kernel function k that can produce a positive definite Gram matrix.

Equation (21) is the same as the kernel Mahalanobis distance measure reported in [18]. It can also be used as a pattern classifier as shown in Fig. 1 for two illustrative toy problems. In Fig. 1, the outputs of the RX-algorithm and kernel RX-algorithm are shown for two different distributions of the data. These two distributions consist of a single Gaussian and a mixture of two Gaussians. As shown in Fig. 1, clearly the kernel RX-algorithm does a better discrimination than the conventional RX-algorithm. The contours in Fig. 1 represent the decision boundaries for different thresholds.

B. Gaussianity Assumption in the Feature Space

Once a kernel function k is chosen the associated feature space is uniquely defined [15]. Our choice of kernel in this paper is the Gaussian RBF kernel and for this kernel the dimensionality of the associated feature space is infinite. Because of the nonlinearity of the mapping function Φ associated with the Gaussian RBF kernel, the question arises whether Gaussianity assumption in the feature space is valid.

It has been shown that in the case of Mercer kernels [15] (the Gaussian RBF kernel being an example of Mercer kernels), after the nonlinear mapping of the input data $\mathbf{x}$, the mapped points $\Phi(\mathbf{x})$ do not spread all over the feature space but occupy a small box with rapidly decaying side lengths, implying that the nonlinear mapping Φ is a smooth function [24], [25]. The elements of the mapped input data in the feature space are obtained from the eigenfunctions of the Mercer kernel weighted by the square root of their corresponding eigenvalues [15]. The eigenvalues of the eigenfunctions of the Gaussian kernel decay very fast. Therefore, the mapped data in the feature space occupy a small compact subspace of the infinite feature space, and the value of the elements of the mapped input vector in the feature space decreases as the dimension is increased. Since Φ is a smooth function and continuous, the topographic ordering of the data in the input space is also preserved in the feature space after the nonlinear mapping [21]. Although the above properties of the feature space are not a valid theoretical proof of the Gaussianity of data in the feature space, they provides sufficient evidence that the mapped data are bandlimited, and we could assume that it has approximately a Gaussian distribution.

The Gaussian distribution assumption in the feature space is equivalent to a non-Gaussian distribution assumption in the original input domain [26], [27]. In fact, it is shown in [26] that a Gaussian distribution in the feature space is (up to normalization) equivalent to a Parzen estimator in the original data space. Thus, implementing the RX-algorithm in the feature space with a Gaussian assumption is equivalent to a nonlinear RX-algorithm in the original input space. Furthermore, if the original input data was Gaussian, then the Gaussian assumption in the feature space using the RBF kernel is still valid, since the nonlinear mapping Φ is a smooth function and the topographic ordering of the input data is preserved in the feature space. In fact, the Gaussian distribution has previously been assumed in the feature space, generating satisfactory experimental results in image segmentation and graphical model learning [18], [26], [27].

V. Implementation of Kernel RX-Algorithm

The kernel matrix $\hat{\mathbf{K}}_b$ can be estimated either globally or locally. The global estimation must be performed prior to detection and normally needs a large amount of data samples to successfully represent all the background types present in a given dataset. In this paper, to globally estimate $\hat{\mathbf{K}}_b$ we need to use all the spectral vectors in a given test image. A well-known data clustering algorithm (k-means [28]) is used on all the spectral vectors in order to generate a significantly less number of spectral vectors (centroids) from which $\hat{\mathbf{K}}_b$ is estimated. By using a small number of distinct background spectral vectors, a manageable kernel matrix is generated where a more efficient kernel RX-algorithm is now implemented.

For local estimation of $\hat{\mathbf{K}}_b$, we use local background samples, which are from the neighboring area of the pixel being tested. For each test pixel location, a dual concentric rectangular window is used to separate a local area into two regions—the inner window region (IWR) and the outer window region (OWR), as shown in Fig. 2; the local kernel matrix and the background covariance matrix are calculated from the pixel vectors in the OWR. The dual concentric windows naturally divide the local area into the potential target region, the IWR, and the background region, the OWR, whose local statistics in the original and nonlinear feature domain are compared using the conventional RX- and kernel RX-algorithms, respectively. The size of the IWR is set to enclose the targets that are to be detected whose approximate size is based on prior knowledge of the range, field of view (FOV), and the dimension of the biggest target in the given dataset. Similarly, the size of the OWR is set to include sufficient statistics from the neighboring background.

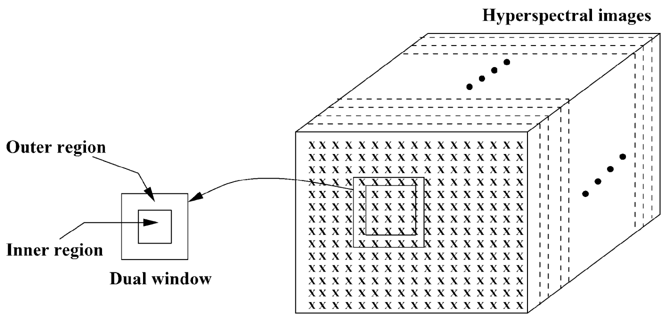

Fig. 2. Example of the dual concentric window in the hyperspectral images.

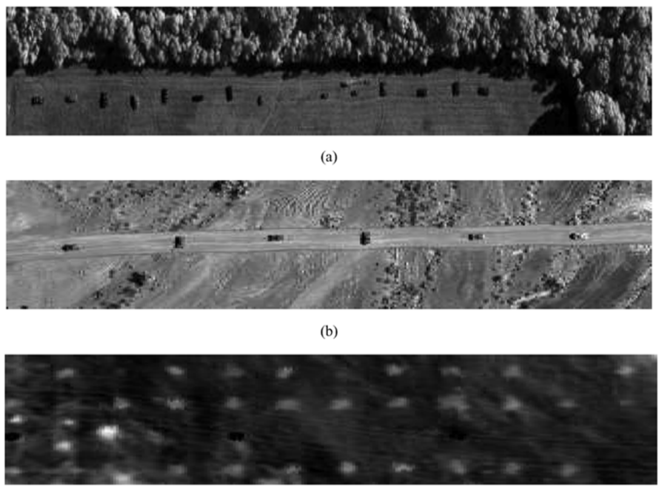

Fig. 3. Sample band images (48th) from HYDICE images and mine image. (a) The Forest Radiance I image, (b) the Desert Radiance II image, and (c) the hyperspectral mine image.

VI. Simulation Results

In this section, we apply both the kernel RX- and conventional RX-algorithms to two Hyperspectral Digital Imagery Collection Experiment (HYDICE) images—the Forest Radiance I (FR-I) image and the Desert Radiance II (DR-II) image—and the hyperspectral mine image, as shown in Fig. 3. FR-I includes 14 targets, and DR-II contains six targets along the road; all the targets are military vehicles. The hyperspectral mine image contains a total of 33 surface mines. A HYDICE imaging sensor generates 210 bands across the whole spectral range (0.4–2.5 μm), but we only use 150 bands by discarding water absorption and low SNR bands; the bands used are the 23rd–101st, 109th–136th, and 152nd–194th. The hyperspectral mine image consists of 70 bands whose spectral range spans 8–11.5 μm.

The Gaussian RBF kernel $k(\mathbf{x},\mathbf{y}) = \exp((-\|\mathbf{x}-\mathbf{y}\|^2)/(c))$ is used to implement the kernel-RX-algorithm. The particular reason why the Gaussian RBF kernel is chosen in this paper

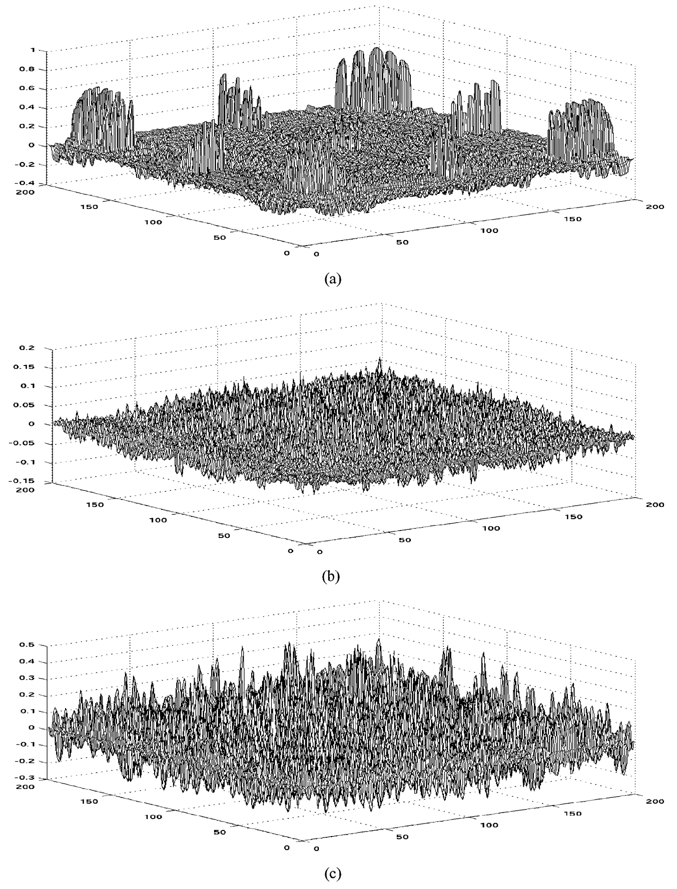

Fig. 4. Example of background kernel matrices $\hat{\mathbf{K}}_b$ obtained from the outer window region when the center of the dual window is located at (a) target, (b) grass, and (c) tree regions in FR-I.

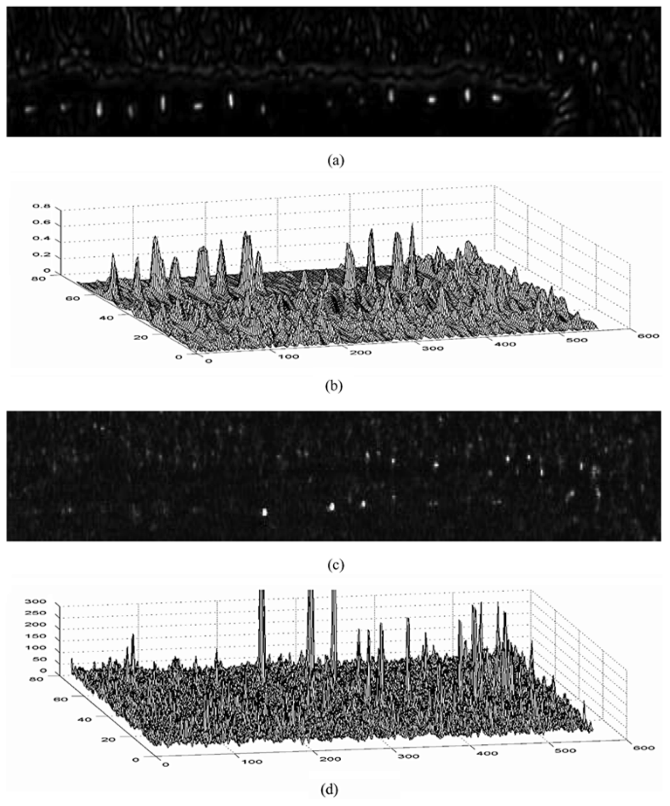

Fig. 5. Detection results for the FR-I image using the kernel RX-algorithm and conventional RX-algorithm based on the local dual window. (a) Kernel RX, (b) 3-D plot of (a), (c) RX, and (d) 3-D plot of (c).

is that it is a *translation-invariant* kernel; it only depends on $\mathbf{x} - \mathbf{y}$, i.e., the difference between $\mathbf{x}$ and $\mathbf{y}$. Therefore, using the Gaussian RBF kernel, the absolute positions of individual vectors are not important. This translation-invariant property is particularly relevant to hyperspectral anomaly detection where local spectral difference between the potential target and its neighboring background regions is compared. Furthermore, for Gaussian RBF kernels its Fourier transform is also Gaussian, implying that the associated nonlinear map is smooth, and its Fourier transform decays rapidly. This property also implies that the mapped data in the feature space occupies a small subspace of the feature space [15]. The choice of c, the width of the Gaussian RBF kernel, is also critical. It should be chosen such that the overall data variations can be fully exploited by the RBF function. In this paper, the value of c was determined experimentally and was set to 40.

All the pixel vectors in each hyperspectral test image are first normalized by a constant, which is a maximum value obtained from all the spectral components of the spectral vectors in the corresponding test image, so that the entries of the normalized pixel vectors fit into the interval of spectral values between zero and one. The rescaling of pixel vectors was mainly performed to effectively utilize the dynamic range of Gaussian RBF kernel. The sizes of the IWR and OWR used for the local kernel and covariance matrix estimations were 5×5 and 15×15 pixel areas, respectively. The size of the OWR was set to include a sufficient number of spectral vectors to generate the kernel matrix $\hat{\mathbf{K}}_b$. For the global kernel matrix estimation, the number of the representative spectral vectors obtained from the k-means procedure was set to 600, which means the number of centroids generated by the k-means was 600.

We used the local dual-window for the local covariance and kernel matrix estimations for the conventional RX- and kernel RX-algorithms, respectively, and the performance between the two algorithms was compared. We also estimated the global kernel matrix for the HYDICE images and the performance between the kernel RX-algorithms implemented with the local and global kernel matrices was compared.

Fig. 4 shows examples of the background kernel matrices (BKM) $\hat{\mathbf{K}}_b$ obtained from the pixels in the OWR when the center of the dual window was located at the target, grass, and tree regions in FR-I. The background kernel matrix around the target region, as shown in Fig. 4(a), contained several groups of large values in contrast to most of the entries which are small in value, indicating that the OWR included a small number of target pixels that were not covered by the IWR. The entries of the grass BKM, as shown in Fig. 4(b), were, in general, small and flat in value, as expected. However, sudden changes in values within the neighboring entries were observed for the tree BKM mainly due to the irregular reflectivity in the tree area.

Figs. 5–7 show the anomaly detection results of both the kernel RX and the conventional RX using the local dual window applied to the FR-I image, DR-II image and the hyperspectral mine image, respectively. The kernel RX detected most of the targets and mines with a few false alarms, while the conventional

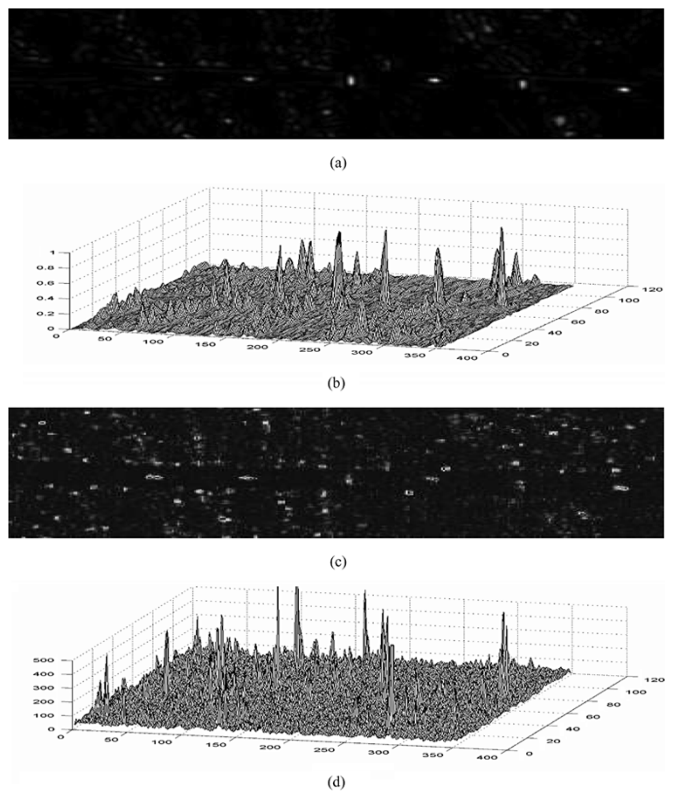

Fig. 6. Detection results for the DR-II image using the kernel RX-algorithm and conventional RX-algorithm based on the local dual window. (a) Kernel RX, (b) 3-D plot of (a), (c) RX, and (d) 3-D plot of (c).

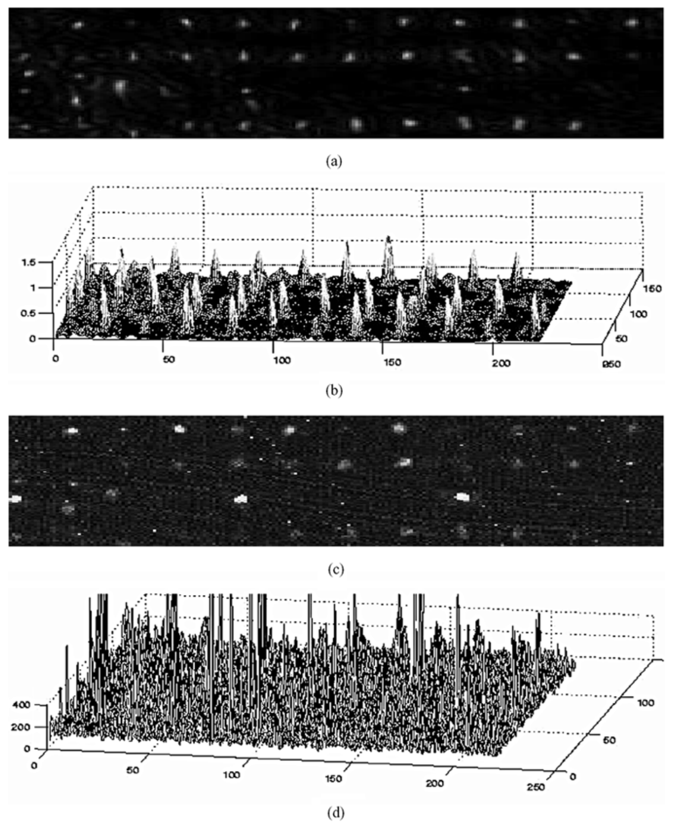

Fig. 7. Detection results for the mine image using the kernel RX-algorithm and conventional RX-algorithm based on the local dual window. (a) Kernel RX, (b) 3-D plot of (a), (c) RX, and (d) 3-D plot of (c).

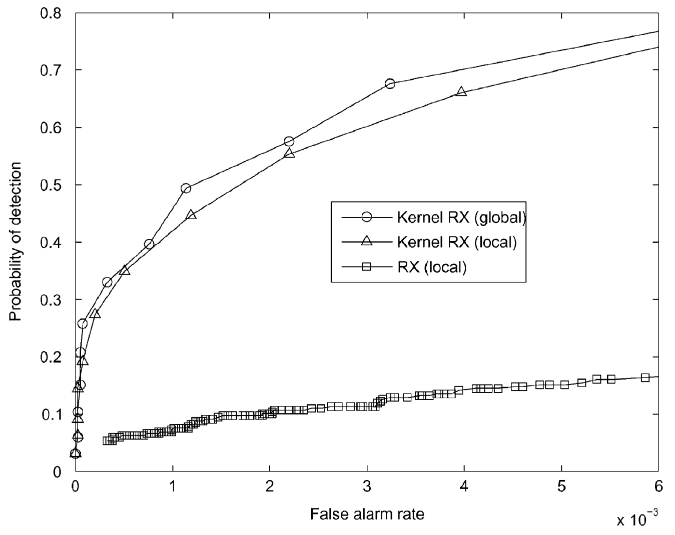

Fig. 8. ROC curves obtained by the kernel RX-algorithm based on the global and local kernel matrices and the conventional RX-algorithm based on the local covariance matrix for the FR-I image.

RX generated much more false alarms and missed some targets; especially, in the case of FR-I the conventional RX missed seven successive targets from the left.

The receiver operating characteristics (ROC) curves representing detection probability P_d versus false-alarm rates N_f were also generated to provide quantitative performance comparison. For ROC curves generation, based on the ground truth information for the HYDICE images and the mine image, we obtain the coordinates of all the rectangular target regions. Every target pixel inside the target regions is then considered as a target candidate to be detected. P_d and N_f are defined as

$$P_d := \frac{N_{\text{hit}}}{N_t} \quad N_f := \frac{N_{\text{miss}}}{N_{\text{tot}}} \tag{22}$$

where N_{hit} represents the number of target pixels detected given a certain threshold; N_t represents the total number of target pixels in the images; N_{miss} represents the number of background pixels detected; and N_{tot} represents the total number of pixels in the images. P_d becomes 1 only when all the individual target pixels within a target are detected; perfect detection is therefore difficult to achieve, and the values of P_d for both the kernel RX and conventional RX are usually less than 1. For both the HYDICE images and the mine image, the kernel RX showed significantly improved performance over the conventional RX.

Figs. 8 and 9 show the ROC curves for the detection results for FR-I and DR-II images, as shown in Figs. 5 and 6, using the kernel RX and the conventional RX based on the local dual window. Figs. 8 and 9 also include the ROC curves for the kernel RX based on the global kernel matrix, as described in Section V. The global method for the kernel RX provided slightly improved performance over the local method for the HYDICE images that were tested. Fig. 10 shows the the ROC curves for the detection results for the hyperspectral mine image, as shown in Fig. 7, using the kernel RX and the conventional RX based on the local dual window. Note that the kernel RX significantly outperformed the conventional RX at lower false-alarm rates.

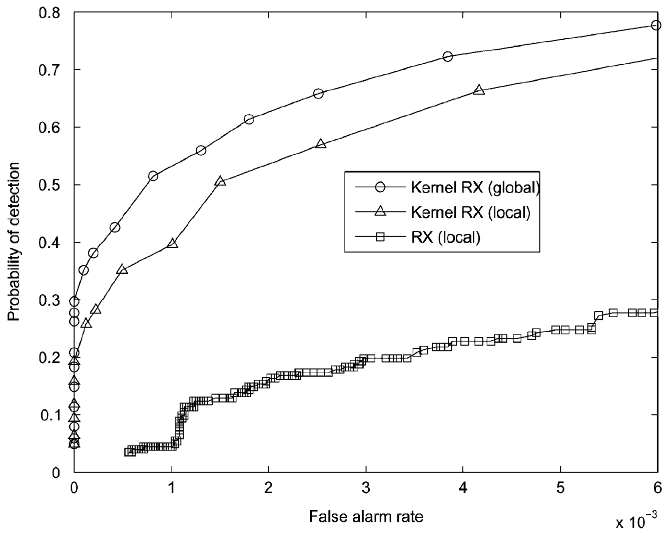

Fig. 9. ROC curves obtained by the kernel RX-algorithm based on the global and local kernel matrices and the conventional RX-algorithm based on the local covariance matrix for the DR-II image.

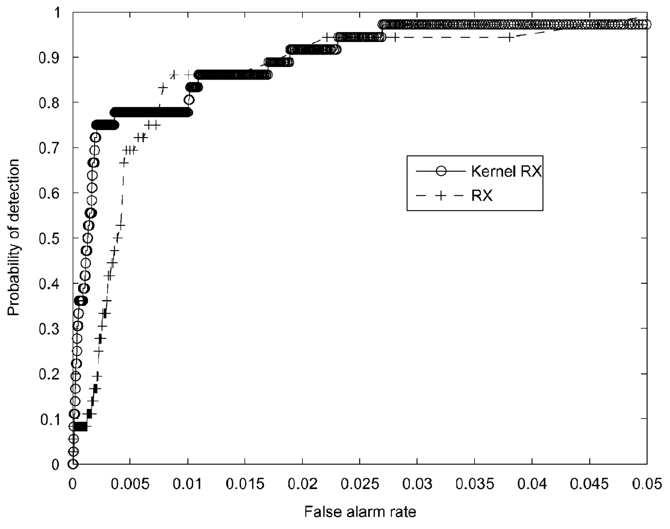

Fig. 10. ROC curves obtained by the kernel RX-algorithm and the conventional RX-algorithm based on the local dual window for the hyperspectral mine image.

VII. Conclusion

We have extended the RX-algorithm in the original input space to a high-dimensional feature space in order to obtain a nonlinear version of the RX-algorithm. The corresponding GLRT expression of the nonlinear RX-algorithm is then kernelized in terms of kernels. Without the kernelization process, the implementation of the nonlinear GLRT is impractical because of the infinite dimensionality of the associated feature space. The GLRT expression of the kernel RX is similar to the conventional RX, but every term in the expression is in kernel form, which can be readily calculated in terms of the input data in its original data space.

The kernel RX-algorithm showed superior detection performance over the conventional RX-algorithm given the two HYDICE and mine hyperspectral images that were tested. The kernel RX-algorithm produced a lot fewer false alarms and detected most of the targets. This is mainly because the high-order correlations between the spectral bands are exploited by the kernel RX. In this paper, only a small number of test images (the HYDICE images and mine image) were used for the performance comparison between the kernel RX-algorithm and conventional RX-algorithm and also for the performance of the kernel RX-algorithm associated with the local and global kernel matrix estimations. However, a larger hyperspectral database is currently being tested in order to generate a more general as well as accurate comparison between the two algorithms.

Appendix I

In this Appendix, derivation of Kernel PCA and its properties providing the relationship between the covariance matrix and the corresponding Gram matrix are presented. Our goal here is to prove expression (20). To drive the Kernel PCA, consider the background clutter covariance matrix in feature space for the centered data $\mathbf{X}_{b_\Phi} = [\Phi_c(\mathbf{x}_1), \Phi_c(\mathbf{x}_2), \ldots, \Phi_c(\mathbf{x}_M)]$

$$\hat{\mathbf{C}}_{b_\Phi} = \mathbf{X}_{b_\Phi}\mathbf{X}_{b_\Phi}^T. \tag{23}$$

The PCA eigenvectors are computed by solving the eigenvalue problem

$$\begin{aligned}\lambda \mathbf{v}_\Phi &= \mathbf{C}_{b_\Phi}\mathbf{v}_\Phi \\ &= \frac{1}{M}\sum_{i=1}^{M}\Phi_c(\mathbf{x}_i)\Phi_c(\mathbf{x}_i)^T\mathbf{v}_\Phi \\ &= \frac{1}{M}\sum_{i=1}^{M}\langle\Phi_c(\mathbf{x}_i), \mathbf{v}_\Phi\rangle\Phi_c(\mathbf{x}_i)\end{aligned} \tag{24}$$

where $\mathbf{v}_\Phi$ is an eigenvector in $\mathcal{F}$ with a corresponding nonzero eigenvalue λ. Equation (24) indicates that any eigenvector $\mathbf{v}_\Phi$ with corresponding $\lambda \neq 0$ are spanned by the input data $\Phi_c(\mathbf{x}_1), \ldots, \Phi_c(\mathbf{x}_M)$—i.e.,

$$\mathbf{v}_\Phi = \sum_{i=1}^{M}\beta_i\Phi_c(\mathbf{x}_i) = \mathbf{X}_{b_\Phi}\boldsymbol{\beta} \tag{25}$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_M)^T$. Substituting (25) into (24) and multiplying with $\Phi_c(\mathbf{x}_n)^T, n = 1, \ldots, M$, yields

$$\begin{aligned}&\lambda\sum_{i=1}^{M}\beta_i\langle\Phi_c(\mathbf{x}_n), \Phi_c(\mathbf{x}_i)\rangle \\ &\quad= \frac{1}{M}\sum_{i=1}^{M}\beta_i\Phi_c(\mathbf{x}_n)\Phi_c(\mathbf{x}_i)\Phi_c(\mathbf{x}_i)^T\sum_{i=1}^{M}\Phi_c(\mathbf{x}_i) \\ &\quad= \frac{1}{M}\sum_{i=1}^{M}\beta_i\left\langle\Phi_c(\mathbf{x}_n), \sum_{j=1}^{M}\Phi_c(\mathbf{x}_j)\langle\Phi_c(\mathbf{x}_j), \Phi_c(\mathbf{x}_i)\rangle\right\rangle, \\ &\qquad\qquad\text{for all } n = 1, \ldots, M.\end{aligned} \tag{26}$$

We denote by $\mathbf{K}_b = \mathbf{K}(X_b, X_b) = (\mathbf{K})_{ij}$ the $M \times M$ kernel (Gram) matrix whose entries are the dot products $\langle \Phi_c(\mathbf{x}_i), \Phi_c(\mathbf{x}_j)\rangle$. Equation (24) can now be rewritten as

$$M\lambda\boldsymbol{\beta} = \mathbf{K}_b\boldsymbol{\beta} \tag{27}$$

where $\boldsymbol{\beta}$ turns out to be the eigenvectors with nonzero eigenvalues of the kernel matrix $\mathbf{K}_b$, as shown in [19]. Note that each $\boldsymbol{\beta}$ needs to be normalized by the square root of its corresponding eigenvalue.

Furthermore, we assumed that the data were centered in the feature space; however, we cannot center the data in the high-dimensional feature space because we do not have any knowledge about the nonlinear mapping Φ. Therefore, we have to start with the original uncentered data and the resulting Gram matrix $\mathbf{K}_b$ needs to be properly centered. As shown in [19], the centered Gram matrix $\hat{\mathbf{K}}_b$ can be obtained from the uncentered Gram Matrix $\mathbf{K_b}$ by

$$\hat{\mathbf{K}}_\mathbf{b} = (\mathbf{K_b} - \mathbf{1}_M\mathbf{K_b} - \mathbf{K_b}\mathbf{1}_M + \mathbf{1}_M\mathbf{K_b}\mathbf{1}_M) \tag{28}$$

where $(\mathbf{1}_M)_{ij} = 1/M$ is an $M \times M$ matrix. From the definition of PCA in the feature space (24) and the Kernel PCA (27), we can now write the eigenvector decomposition of the background covariance matrix and Gram matrix as

$$\hat{\mathbf{C}}_{b_\Phi} = \mathbf{V}_\Phi\boldsymbol{\Lambda}_b\mathbf{V}_\Phi^T \tag{29}$$

$$\hat{\mathbf{K}}_b = \mathcal{B}\Omega_{\mathbf{K}_b}\mathcal{B}^T \tag{30}$$

respectively. Using pseudoinverse matrix properties [29], the pseudoinverse background covariance matrix $\hat{\mathbf{C}}_{b_\Phi}^{\#}$ and inverse Gram matrix $\hat{\mathbf{K}}_b^{-1}$ can also be written as

$$\hat{\mathbf{C}}_{b_\Phi}^{\#} = \mathbf{V}_\Phi\boldsymbol{\Lambda}_b^{-1}\mathbf{V}_\Phi^T \tag{31}$$

$$\hat{\mathbf{K}}_b^{-1} = \mathcal{B}\Omega_{\mathbf{K}_b}^{-1}\mathcal{B}^T \tag{32}$$

respectively. From the relationship between the eigenvalues of the covariance matrix in the feature space and the Gram matrix described in (27), we have

$$\boldsymbol{\Lambda}_b = \frac{1}{M}\Omega_{\mathbf{K}_b} \tag{33}$$

where $\boldsymbol{\Lambda}_b$ is a diagonal matrix with its diagonal elements being the eigenvalues of $\hat{\mathbf{C}}_{b_\Phi}$, and $\Omega_{\mathbf{K}_b}$ is a diagonal matrix with diagonal values equal to the eigenvalues of the Gram matrix $\hat{\mathbf{K}}_b$. Substituting (33) into (32), we obtain the relationship

$$\hat{\mathbf{K}}_b^{-1} = \frac{1}{M}\mathcal{B}\boldsymbol{\Lambda}_b^{-1}\mathcal{B}^T \tag{34}$$

where M is a constant representing the total number of background clutter samples that can be ignored.

References

[1] D. Manolakis and G. Shaw, "Detection algorithms for hyperspectral imaging applications," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 29–43, Jan. 2002.

[2] J. C. Harsanyi and C.-I. Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 779–785, Jul. 1994.

[3] I. S. Reed and X. Yu, "Adaptive multiple-band CFAR detection of an optical pattern with unknown spectral distribution," *IEEE Trans. Acoust., Speech Signal Process.*, vol. 38, no. 10, pp. 1760–1770, Oct. 1990.

[4] C.-I. Chang, X.-L. Zhao, M. L. G. Althouse, and J. J. Pan, "Least squares subspace projection approach to mixed pixel classification for hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 36, no. 3, pp. 898–912, May 1998.

[5] G. Healey and D. Slater, "Models and methods for automated material identification in hyperspectral imagery acquired under unknown illumination and atmospheric conditions," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 6, pp. 2706–2717, Nov. 1999.

[6] L. L. Scharf and B. Friedlander, "Matched subspace detectors," *IEEE Trans. Signal Process.*, vol. 42, no. 8, pp. 2146–2157, Aug. 1994.

[7] S. M. Schweizer and J. M. F. Moura, "Efficient detection in hyperspectral imagery," *IEEE Trans. Image Process.*, vol. 10, no. 4, pp. 584–597, Apr. 2001.

[8] D. W. J. Stein, S. G. Beaven, L. E. Hoff, E. M. Winter, A. P. Schaum, and A. D. Stocker, "Anomaly detection from hyperspectral imagery," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 58–69, Jan. 2002.

[9] H. Kwon, S. Z. Der, and N. M. Nasrabadi, "An adaptive unsupervised segmentation algorithm based on iterative spectral dissimilarity measure for hyperspectral imagery," *Proc. SPIE*, vol. 4310, pp. 144–152, Jan. 2001.

[10] D. W. J. Stein, "Stochastic compositional models applied to subpixel analysis of hyperspectral imagery," *Proc. SPIE*, vol. 4480, pp. 49–56, Jul. 2001.

[11] H. Kwon, S. Z. Der, and N. M. Nasrabadi, "Adaptive anomaly detection using subspace separation for hyperspectral images," *Opt. Eng.*, vol. 42, no. 11, pp. 3342–3351, Nov. 2003.

[12] C.-I. Chang and S.-S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 40, no. 6, pp. 1314–1325, Jun. 2002.

[13] X. Yu and I. S. Reed, "Comparative performance analysis of adaptive multispectral detectors," *IEEE Trans. Signal Process.*, vol. 41, no. 8, pp. 2639–2656, Aug. 1993.

[14] C. M. Stellman, G. G. Hazel, F. Buchholtz, and J. V. Michalowicz, "Real-time hyperspectral detection and cuing," *Opt. Eng.*, vol. 39, no. 7, pp. 1928–1935, Jul. 2000.

[15] B. Schökopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA: The MIT Press, 2002.

[16] J. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Face recognition using kernel direct discriminant analysis algorithm," *IEEE Trans. Neural Networks*, vol. 14, no. 1, pp. 117–126, Jan. 2003.

[17] P. P. Pekalska and E. R. P. W. Duin, "A generalized kernel approach to dissimilarity-based classification," *J. Mach. Learn.*, vol. 2, pp. 175–211, 2001.

[18] A. Ruiz and E. Lopez-de Teruel, "Nonlinear kernel-based statistical pattern analysis," *IEEE Trans. Neural Networks*, vol. 12, no. 1, pp. 16–32, Jan. 2001.

[19] B. Schökopf, A. J. Smola, and K.-R. Müller, "Kernel principal component analysis," *Neural Comput.*, no. 10, pp. 1299–1319, 1999.

[20] G. Baudat and F. Anouar, "Generalized discriminant analysis using a kernel approach," *Neural Comput.*, no. 12, pp. 2385–2404, 2000.

[21] M. Girolami, "Mercer kernel-based clustering in feature space," *IEEE Trans. Neural Networks*, vol. 13, no. 3, pp. 780–784, May 2002.

[22] H. Kwon and N. M. Nasrabadi, "Hyperspectral target detection using kernel matched subspace detector," in *Proc. IEEE Int. Conf. Image Processing*, Singapore, Oct. 2004, pp. 3327–3330.

[23] I. T. Joliffe, *Principal Component Analysis*. Berlin, Germany: Springer-Verlag, 1986.

[24] R. C. Williamson, A. J. Smola, and B Schölkopf, "Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators," *IEEE Trans. Inf. Theory*, vol. 47, no. 6, pp. 2516–2532, Sep. 2001.

[25] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *J. Mach. Learn. Res.*, vol. 47, pp. 1–48, 2002.

[26] D. Cremers, T. Kohlberger, and B. Schölkopf, "Shape statistics in kernel space for variational image segmentation," *Pattern Recognit.*, vol. 36, pp. 1929–1943, 2003.

[27] F. R. Bach and M. I. Jordan, "Learning graphical models with mercer kernels," *Adv. Neural Inf. Process. Syst.*, vol. 15, 2003.

[28] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.

[29] G. Strang, *Linear Algebra and Its Applications*. Orlando, FL: Harcourt Brace, 1986.

Heesung Kwon (S'96–M'99) received the B.Sc. degree in electronic engineering from Sogang University, Seoul, Korea, in 1984, and the M.S. and Ph.D degrees in electrical engineering from the State University of New York, Buffalo, in 1995 and 1999, respectively.

From 1983 to 1993, he was with Samsung Electronics Corporation, where he worked as an engineer. Since 1996, he has been with the U.S. Army Research Laboratory, Adelphi, MD. His interests include hyperspectral image analysis, pattern recognition, statistical learning, and image/video compression.

Nasser M. Nasrabadi (S'80–M'84–SM'92–F'01) received the B.Sc. (Eng.) and Ph.D. degrees in electrical engineering from the Imperial College of Science and Technology, University of London, London, U.K., in 1980 and 1984, respectively.

From October 1984 to December 1984, he was with IBM (UK) as a Senior Programmer. From 1985 to 1986, he was with the Philips Research Laboratory in New York as a member of the technical staff. From 1986 to 1991, he was an Assistant Professor in the Department of Electrical Engineering, Worcester Polytechnic Institute, Worcester, MA. From 1991 to 1996, he was an Associate Professor with the Department of Electrical and Computer Engineering, State University of New York, Buffalo. Since September 1996, he has been a Senior Research Scientist with the U.S. Army Research Laboratory, Adelphi, MD, working on image processing and automatic target recognition. His current research interests are in image and video compression, packet video, automatic target recognition, and neural networks applications to image processing.

Dr. Nasrabadi has served as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS, SYSTEMS, and VIDEO TECHNOLOGY, and the IEEE TRANSACTIONS ON NEURAL NETWORKS. He is a Fellow of SPIE.